SF4 Handout 2 – System Design

1.0 INTRODUCTION

Once you have completed the introductory handout, which should be accomplished within the first couple of sessions, you should spend about a week working on your initial design, both in and out of the timetabled sessions. This should include:

- Specifying an application for your system (e.g. are you going to build a weather station, guitar tuner etc.).
- Producing a conceptual overall design (i.e. what data rates and formats are involved, what MCU features will be needed).
- Producing a detailed hardware design including circuit diagrams and a component list.
- Producing an approximate firmware flow diagram and decide how the Arduino will interact with the computer and the user.

This should all be included in your interim report, and components should be ordered by the interim report deadline at the latest. The interim report deadline is given in the main project handout.

You should then spend the next two weeks focusing on detailed hardware, firmware and software implementation, as well as system integration and testing. This will all take a considerable amount of time, and should be started as early as possible. In particular, you can start working on the firmware/software before the hardware is ready, and should leave time at the end of the project to test and perfect the whole system.

2.0 INITIAL DESIGN

2.1 Application

You are encouraged to come up with a creative application for your system. The only restrictions are that it should demonstrate two-way communication between the MCU and the hardware/sensors and also the MCU and the PC and that it should be a prototype of useful product with a real application.

The marking criteria are contained in the main project handout. However, broadly you will receive credit for how creative/useful your idea is, how technically challenging it is, how well it is implemented and how well it is communicated in your report and demonstration.

You are encouraged to pick a concept that can be initially implemented and tested in a relatively simple form, but which can then be extended with more advanced features. This will help with time and risk management.

Your system's application will obviously be constrained by what is possible in practice, and thus you will need to think about the issues in sections 2.2 and 2.3 when considering possible applications.

Your design process will inevitably be iterative, and you may later come back and modify your application based on experience in the conceptual design and/or implementation stages.

2.2 Conceptual Design

Once you have selected an application that you think is realistic and allows enough scope to develop an interesting product, you will need to think about the overall conceptual design. Issues to consider include:

What sensors are available and within budget?

You can search the Farnell and RS websites for components including sensors. Typical sensors might include microphones, temperature sensors, other environmental sensors such as pressure/humidity sensors, light sensors, IR sensors, distance sensors, magnetometers etc. Many other sensors are also available, including accelerometers, gyroscopes etc. However, be careful to check the packages that the sensor IC is available in – some are only available as very small surface-mount components that are difficult, although not impossible, to incorporate on a breadboard.

What other hardware is required?

You will probably need to amplify and filter analogue signals, and may also wish to include feedback or control such as automatic gain control, multiplexing, digital or analogue outputs, displays, battery power supplies etc. Suitable circuits can often be found online (and should be referenced in your reports where that's the case) or sometimes in your lecture notes. Components can, again, be found on the Farnell or RS websites.

What MCU features will be needed?

The Arduino website https://www.arduino.cc/en/main/arduinoBoardUno
provides an overview of its features and various details on their use (see also the additional reading material provided on the Moodle website for more details and more advanced features). In addition to the CPU, which executes the firmware, it has a number of peripherals that can perform other functions. You will need to use some of these peripherals, but the exact combination will depend on your application.
You should choose which of the Arduino features will be needed, ensuring that the resulting firmware can realistically be implemented in the time available.

What data rates, memory sizes, power requirements etc are involved?

Consider the rate at which you are sampling, storing and transferring data. Remember that communication with the PC is limited (depending on your exact design choices) and that you have a limited amount of on-chip RAM.

Consider also how much current your circuit is likely to draw and whether it's within the limit that the Arduino board can provide. If not you may need to use a bench-top power supply instead of the USB supply. You could also consider a battery supply for portability.

What software interface and processing will be implemented?

You should plan to produce a nice, user-friendly interface to control your product and display its data/results (interfacing with python or matlab is an option via their serial port read/write capabilities, see introductory handout). Think about what it will look like, how the user will interact with it and

what communication messages will need to be defined to allow the required communication between PC and MCU.

You should also aim to implement processing that builds on the basic sensors to perform an interesting function or functions. For example a microphone can be implemented relatively easily, and firmware can be written to sample the sound waveform and send it to the PC. However, to turn that into a basic guitar tuner (for example) you would need to implement an FFT (Fast Fourier Transform) and compare the detected frequency with one that you're trying to tune to. More advanced extensions could then be considered.

What are the tradeoffs between hardware, firmware and software?

You will often have a choice between implementing features in hardware, firmware or software. For example, if you want to detect the magnitude of a signal from a microphone to indicate audio volume, you could use a peak detector circuit in hardware and sample the output of that at a low rate. Alternatively you could sample the amplified microphone output directly at a high rate. You could then detect the peak in firmware or send all the data to the PC and detect the peak in software. There is usually no right answer, but different tradeoffs are involved and you should consider the alternatives before making a decision. The choice should also be justified in your reports.

How will you protect the MCU, PC etc?

Your design may risk damage to the MCU. Make sure each of the input/output pins of the board are used according to their specifications in terms of the rated voltage and current. Details for these ratings can be found at https://www.arduino.cc/en/main/arduinoBoardUno . For example each of the 14 digital pins operates at 5 V and can provide or receive 20 mA as a recommended operating condition. Exceeding 40 mA would cause a permanent damage to the board, so you might consider using fuses to avoid this.

2.3 Detailed Hardware Design

You will need to consider your hardware design while choosing an application and during the conceptual design stage. Once you have a conceptual design you should finalise your hardware design, which should be included in the interim report. This should include all necessary circuits, as well as power supplies, generation of single- or double-sided power supplies for op-amps, voltage level compatibility, decoupling, protection etc.

2.4 Component Order

You should order components by the interim report deadline at the latest. Earlier ordering of components will allow you to spend more time working on the design later on.

To order components, you should:

1. Check that the components are in stock with Farnell or RS (and don't need to be sent from US stock).
2. Check the datasheets for all your components to ensure they're compatible with your circuit and with each other (in terms of power supply voltages, logic levels, data rates, external

circuit requirements, component package sizes etc). Discuss any issues with the demonstrators

3. Fill out the component order form provided.
4. Check your final selection with the demonstrators
5. Submit your final order form to the lab technicians.

Additional components can be ordered later on, but you should try and ensure you have all the components you'll need available as soon as possible, including any passive components such as capacitors that may not be available in the lab.

Note that it is the cost of components included in your final circuit that counts towards your budget, not the total amount spent on components.

Components should arrive within one (or occasionally two) working day(s). **If your component order does not arrive when you expect it to, check immediately with the lab technicians and discuss the issue with the demonstrators at the next available opportunity.** If your components are delayed or don't arrive due to an administrative error, the components should be re-ordered as soon as possible. Doing this promptly will save you a lot of time waiting for components. While waiting for components you can of course continue to work on the firmware and software parts of the project.